Remarks

Status of Claims

Claims 1-30 are pending in the application. Claims 1, 17, 21, and 22 are in independent form. Claims 1-21 and 27-30 are allowed. The final Office action rejected claims 22-26 as being anticipated by U.S. Patent Publication No. 2002/0007461 ("Garrison").

Prior Art Rejection

Brief Summary of Garrison

To better understand the distinctions between Garrison and the claims in the Present Application, Garrison will be briefly summarized. Garrison generally relates to securing remote access to a database and utilizes a client computer (client), a server computer (server), and a database system having data stored in plaintext. To start a data session, the client establishes communication with the server and the server generates and transmits a new encryption key for the current data session to the client. Upon receiving the new encryption key the client uses it to encrypt data communicated by the client during the reminder of the data session.

After receiving the new encryption key, the client encrypts the user's password and log name with the new encryption key and transmits them to the server. The server receives the user password and log name and decrypts them using the new encryption key. The server also translates the user password into an alias or different password. When the client submits a request for data contained in the database system, the server accesses the database system using the alias password. The database system allows the server to access information within the database system based on the alias password. Since the database system recognizes the alias password instead of the user password, only attempts to access the database via the server (after passing the security measures in place at the server) are successful.

Claim 22

Claim 22 reads as follows (with emphasis added):

> **22.** A method for accessing data stored in a secure database comprising:
>
> receiving a request for data, the data stored in an obfuscated format within a data crystal;
>
> determining an accessible predefined query based upon query permissions stored within a key crystal; and
>
> calling the accessible predefined query to direct an iterator to access, deobfuscate, and return data satisfying the request so that the data stored within the data crystal remains secure.

Claim 22 was rejected under section 102(e) as allegedly being anticipated by Garrison. The Applicant respectfully traverses and asserts that claim 22 is patentable over Garrison for at least the following reasons.

GARRISON DOES NOT DISCLOSE
DATA STORED IN AN OBFUSCATED FORMAT

The Office action cites paragraphs **[0022]**, **[0040]**, and **[0045]** of Garrison to support the allegation that data is stored in an obfuscated format. These paragraphs do not support such an allegation. In fact, Garrison does not appear to discuss data stored in an obfuscated format within a data crystal at all. At best the server in Garrison encrypts plaintext data retrieved from the database system before transmitting it to the client. See paragraph **[0077]** of Garrison.

Paragraph **[0022]** indicates that access to Garrison's remote database system remains secure even if a hacker intercepts and deciphers encrypted messages transmitted between the client and the server. Apparently, by using an alias password, Garrison's remote database system is immune to decrypting an encrypted password. This paragraph does not talk about the data in the database and does not indicate that the data is stored in an obfuscated format within a data crystal. In fact, paragraph **[0072]** seems to indicate that plaintext data is stored within the database.

Paragraph **[0040]** discusses an alternative to encrypting the new encryption key with a public encryption key supplied by the client before sending the new encryption key to the client. Instead of using a public encrypting key, a standard algorithm may be used by the server before sending the new encryption key to the client. By way of example, the server transmits a plurality of encryption keys along with an index indicating which of the

keys is the new encryption key for the data session. The client can process the index via the standard algorithm to determine which is the new encryption key. In other words, Garrison speaks of the new encryption key being used to decrypt data sent from the server (after the plaintext data has been retrieved from the database system).

Paragraph **[0045]** discloses that the client encrypts its request for data using the new encryption key before sending the request for data to the server. This paragraph does not discuss the data being stored in an obfuscated format within a data crystal. For at least these reasons, claim 22 and its respective dependent claims are not anticipated by Garrison.

GARRISON DOES NOT DISCLOSE
DETERMINING AN ACCESSIBLE PREDEFINED QUERY
BASED ON QUERY PERMISSIONS STORED WITHIN A KEY CRYSTAL

Claim 22 refers to determining an accessible predefined query based upon query permissions stored within a key crystal. The Office action alleges that paragraphs **[0045]**, **[0046]**, and **[0047]** of Garrison anticipates this portion of the claim. The Applicant respectfully traverses.

By way of example, and not limitation, Figure 9 of the Present Application illustrates a process that could occur at a customer's site when using a crystal database including a crystal set 108. The crystal set 108 contains all the individual data crystals sent to the customer, including active crystals 124 and inactive crystals 126. The active crystals 124 are crystals to which the customer has valid rights to access. The inactive crystals 126, although complete with respect to the information they contain, are crystals to which the customer does not have valid rights to access. A key crystal 110, which is supplied to each customer, can contain crystal permission information 112 as well as query permission information 114. The crystal permission information 112 indicates which crystals are active and which crystals are inactive for that individual customer. Similarly, the query permission information 114 stores information pertaining to which of the predefined queries 34a and 34b the individual customer is licensed or permitted to use.

The database customer may be provided with a crystal database containing both active crystals 124 and inactive crystals 126, as well as permissible queries 34a and 34b and impermissible queries (not shown). If the customer wishes to access the data records within the inactive crystals 126 or to gain permission to use impermissible queries, the database designer can send the customer a new key crystal with updated crystal permission

information 112 and updated query permission information 114. The new key crystal would contain information permitting the customer to use the newly-permitted queries and newly-active crystals when the new key crystal is accessed by an iterator. In this manner, the customer can quickly change the type of information to which he has access without replacing the entire database.

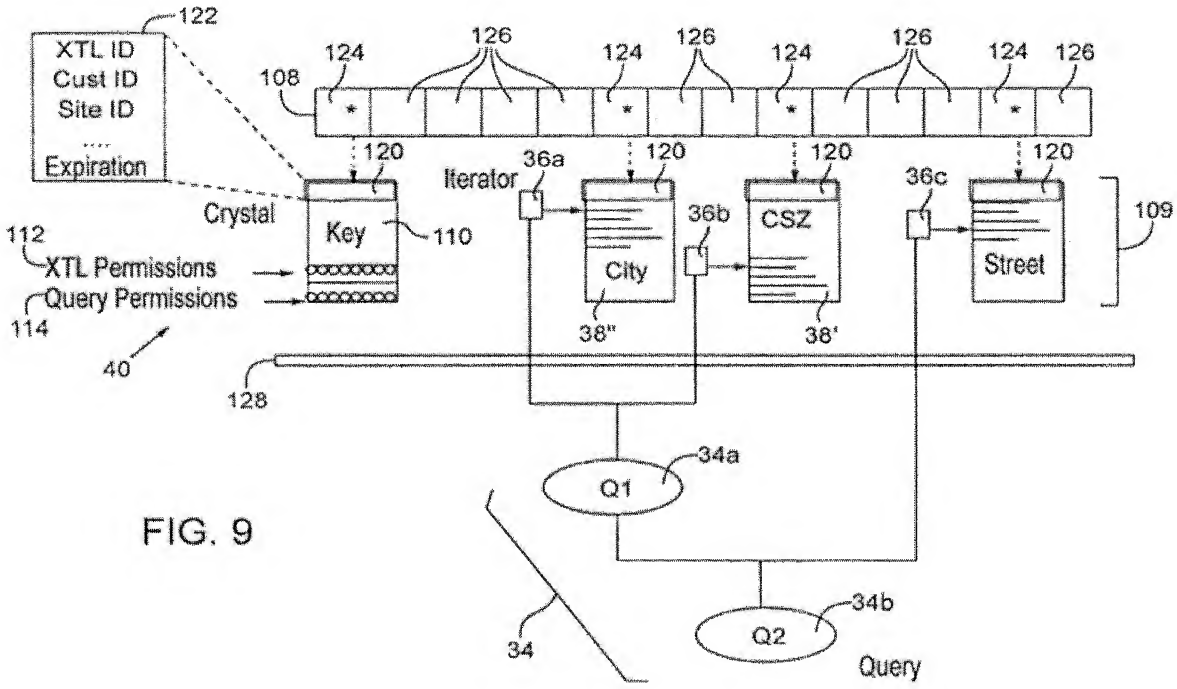

FIG. 9

FIG. 9 of the Present Application

Paragraphs **[0045]**, **[0046]**, and **[0047]** of Garrison essentially discuss the client encrypting a request for data and sending the encrypted request for data to the server. Based on the request, the server determines whether the information requested is accessible to the user by checking entries in a security data table. The request for data may be a code word known to the server. Using the code word eliminates the need for encrypting the request because a hacker that intercepts the code word would likely be unable to extract any useful information therefrom. This is not the same as determining an accessible predefined query based upon query permissions stored within a key crystal for several reasons.

First, there is no indication that the client in Garrison checks query permissions stored within a key crystal to determine accessible predefined queries. Instead, Garrison indicates that the request for data can be any type of query, such as a structured query language (SQL) query. In contrast, claim 22 refers to a predefined query. By way of

example, and not limitation, according to one embodiment, a predefined query does not support a general query language such as that used in structured query language applications (*e.g.*, SQL). See the Present Application at paragraphs **[0011]** and **[0039]**.

Next, paragraphs **[0045]**, **[0046]**, and **[0047]** of Garrison describe checking whether the information requested by the client (*e.g.*, the data) is accessible to the user, not whether a predefined query is accessible. In contrast, claim 22 refers to determining an accessible predefined query. As discussed in more detail below, the predefined query directs an iterator to access, deobfuscate, and return data satisfying the request so that the data stored within the data crystal remains secure. In other words, before accessing any data, a predefined query must first be accessible. For at least these reasons, claim 22 and its respective dependent claims are not anticipated by Garrison.

GARRISON DOES NOT DISCLOSE
A PREDEFINED QUERY DIRECTING
AN ITERATOR TO ACCESS DATA

Claim 22 refers to calling the accessible predefined query to direct an iterator to access data satisfying the request so that the data stored within the data crystal remains secure. The Office action alleges that paragraphs **[0034]** and **[0041]** of Garrison anticipates this portion of the claim. The Applicant respectfully traverses.

By way of example, and not limitation, in the present case:

> database customer applications call queries belonging to the predefined query types to instruct the iterator to access the data records in the database. The database customer can be given access to select predefined query types, and the calling of the queries can be done by the customer application itself as part of standard operations. Database customer applications are only allowed to interact with queries. They cannot interact directly with the iterators. This prevents a customer from using the iterators to extract the entire contents of the crystalized database.

Present Application at paragraph **[0011]**. In other words, access to the iterator is indirect – via the predefined query. By way of example, and not limitation, if a customer has direct access to the query (*e.g.*, queries 34a and 34b), the data is still secure, but if the customer has direct access to the iterator (*e.g.*, iterators 36a-36c), the data is not secure. See the Present Application at paragraph **[0051]**. Paragraphs **[0034]** and **[0041]** of Garrison do not disclose calling the accessible predefined query to direct an iterator to access data satisfying the request so that the data stored within the data crystal remains secure. In fact, paragraphs **[0034]** and **[0041]** of Garrison do not discuss an iterator accessing data nor a

predefined query directing the iterator to access data. Instead, paragraph **[0034]** appears to discuss the hardware of the server. Paragraph **[0041]** of Garrison further discusses the alternative to encrypting the new encryption key with a public encryption key supplied by the client before sending the new encryption key to the client (see paragraph **[0040]** of Garrison). Paragraph **[0041]** indicates that the index could be a code word indicating the placement of the new key within the plurality of keys (*e.g.,* the new key will be the tenth key transmitted by the server). Accordingly, neither paragraphs **[0034]** and **[0041]** of Garrison disclose calling the accessible predefined query to direct an iterator to access data satisfying the request so that the data stored within the data crystal remains secure. For at least these reasons, claim 22 and its respective dependent claims are not anticipated by Garrison.

GARRISON DOES NOT DISCLOSE
A PREDEFINED QUERY DIRECTING
AN ITERATOR TO DEOBFUSCATE DATA

Claim 22 refers to a predefined query directing an iterator to deobfuscate data satisfying the request so that the data stored within the data crystal remains secure. The Office action alleges that paragraphs **[0036]**, **[0037]**, **[0039]**, and **[0047]** of Garrison anticipates this portion of the claim. These paragraphs appear to discuss decrypting the new encryption key and other data transmitted between the client and the server. As previously discussed, Garrison does not discuss data stored in an obfuscated format within a data crystal. If the data stored in Garrison's database is plaintext, there can be no data to deobfuscate. For at least this reason, claim 22 and its respective dependent claims are not anticipated by Garrison.

Claim 24

Claim 24 reads as follows (with emphasis added):

> **24.** The method of claim 22 wherein the request is received from a parser application of an automated data capture and perfection system.

By way of example, and not limitation, an automated data capture and perfection system might obtain data defining a United States postal address from a postal envelope and then compare the obtained address to a known reference address stored in the crystal database 40 in order to identify an illegible city name in the address. In Figure 12 of the Present Application, the data comprising the address is supplied to a customer application

that does data record parsing. The parser application 218 divides the data record address into pieces expected to correspond to data fields. The predefined queries 34 are then called to direct iterators 36 to request data from the crystal database 40. For example, if both a city name and a zip code are distorted in a postal address, and the city appears to begin with the letters "RED," a predefined query 34 can be called to ask for a list of all U.S. cities that start with the letters "RED." It could also be defined to ask for cities that appear to start with the letters "RED" or contain the letters "RED." Once the results are obtained from the crystal database 40, a predefined query 34 can also be used to verify or determine possibilities for the distorted zip code. If the city name data and zip code data are connected with pointers, one of the query results for the possible city names should match with one of the query results for the zip code. The correct result can then be sent in any of numerous forms of output. Examples of output include a corrected address 222, routing information for incoming mail 224, or validated check information 226. The output can also be structured however the database customer prefers, and may depend on the original source or form of the data.

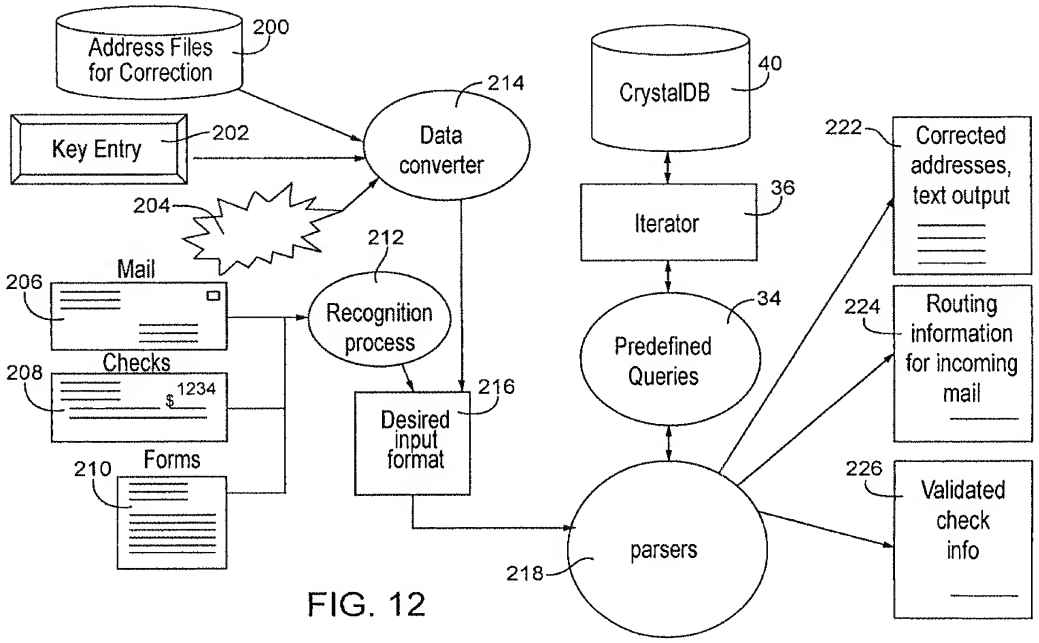

FIG. 12

FIG. 12 of the Present Application

The Office action cites paragraph **[0013]** of Garrison and argues it is inherent that the client referred to in paragraph **[0013]** can run on an application of an automated data capture and perfection system. Inherency may not be established by probabilities or

possibilities – the mere fact that a certain thing may result from a given set of circumstances is not sufficient. See *In re Robertson*, 169 F.3d 743, 745 (Fed. Cir. 1999) (to establish inherency, the extrinsic evidence must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill). See also MPEP § 2112. Garrison does not disclose a parser application of an automated data capture and perfection system making a request for data. For at least this reason, claim 24 is not anticipated by Garrison.

Conclusion

In view of the foregoing, Applicant submits that all claims are in condition for allowance. Therefore issuance of the Notice of Allowance is respectfully requested. The Examiner is welcome to call the undersigned to discuss any aspect of this application.

Respectfully submitted,

RAF Technology, Inc.

By _____
Nathan Scherer
Registration No. 58,460

STOEL RIVES LLP
900 SW Fifth Avenue, Suite 2600
Portland, OR 97204-1268
Telephone: (503) 224-3380
Facsimile: (503) 220-2480